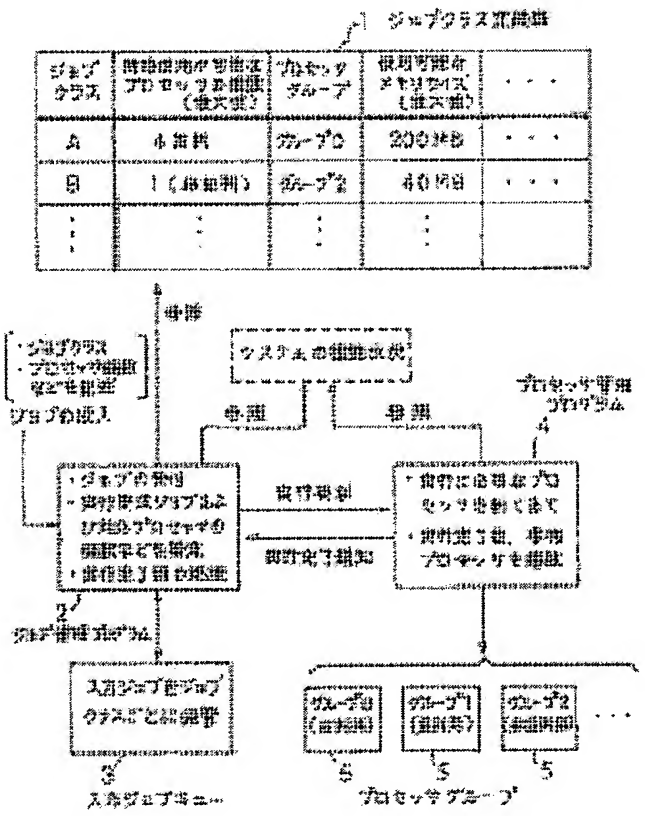
PROCESSOR ALLOCATION SYSTEM

Publication number: JP7200496 (A)
Publication date: 1995-08-04
Inventor(s): OSADA KAZUHISA; SAKAMOTO YOSHINORI
Applicant(s): FUJITSU LTD
Classification:
- international: *G06F15/16; G06F9/46; G06F15/177;* **G06F15/16; G06F9/46;** (IPC1-7): G06F15/16; G06F9/46
- European:
Application number: JP19930337840 19931228
Priority number(s): JP19930337840 19931228

Also published as:
JP3541212 (B2)

Abstract of **JP 7200496 (A)**

PURPOSE:To improve the flexibility of multiprocessor operation by lightening the burden on a job defining person when parallel operating processors are indicated in the case of job execution. CONSTITUTION:A job class definition table 1 contains the number (maximum value) of processors which can use jobs of respective job classes at the same time, a processor group corresponding to it, etc., in advance, and the job defining person only indicates a job class and the number of processors required to execute the job; ; and then a job management program 2 finds the number of processors and processor group corresponding to the requested job to be executed by referring to the job class definition table 1 and informs a processor management program 4 of them, so that the processor management program 4 allocates the processors on the basis of the reported information, system operation state, etc. Further, the processors constituting the processor group are changed and the group in the job class that plural processor groups are made to correspond to is selected dynamically according to the operation state.



(19)日本国特許庁（ＪＰ）　(12) **公開特許公報**（Ａ）　(11)特許出願公開番号

特開平7－200496

(43)公開日　平成７年(1995) ８月４日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ　15/16	３８０　Ｚ			
9/46	３６０　Ｂ	7629－5B		

審査請求　未請求　請求項の数５　ＯＬ　（全９頁）

(21)出願番号　特願平5－337840

(22)出願日　平成５年(1993)12月28日

(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中1015番地

(72)発明者　長田　和久
静岡県静岡市伝馬町16番地の３　株式会社富士通静岡エンジニアリング内

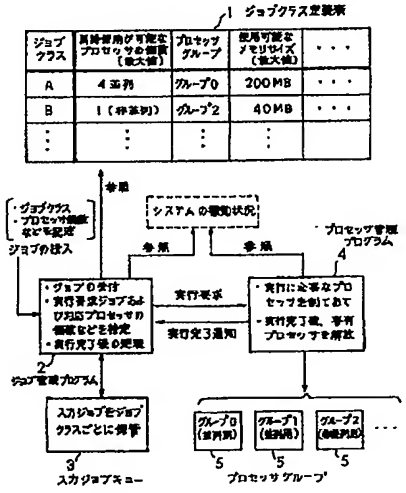
(72)発明者　坂本　喜則
神奈川県川崎市中原区上小田中1015番地
富士通株式会社内

(74)代理人　弁理士　長谷川　文廣　（外２名）

(54)【発明の名称】　プロセッサ割当て方式

(57)【要約】
【目的】　ジョブ実行の際の並列動作プロセッサ指示についてのジョブ定義者の負担を軽くし、マルチプロセッサ運用面での柔軟性を高めることを目的とする。
【構成】　ジョブクラス定義表１に各ジョブクラスのジョブが同時使用できるプロセッサ個数（最大値）やこれに対応したプロセッサグループなどをあらかじめ記載しておき、ジョブ定義者はジョブクラスやジョブの実行に必要なプロセッサ個数を指示すれば、後はジョブ管理プログラム２がジョブクラス定義表１を参照することにより実行要求ジョブに対応のプロセッサ個数やプロセッサグループを求めてプロセッサ管理プログラム４に通知し、プロセッサ管理プログラム４は通知情報とシステム稼働状況などに基づいてプロセッサを割り当てる。また、プロセッサグループを構成するプロセッサの変更や、複数のプロセッサグループを対応させたジョブクラスにおけるグループ選択も稼働状況に応じて動的に行う。

本発明の基本構成



【特許請求の範囲】
【請求項１】　ジョブの定義により少なくとも実行プログラム名およびジョブクラスを記述した各ジョブの投入に対し、ジョブ管理プログラムは、これらの入力ジョブのそれぞれを入力ジョブキューに保管し、かつ実行優先度の高い入力ジョブを選択して当該入力ジョブの実行要求をその実行の際の必要プロセッサ数とともにプロセッサ管理プログラムに通知し、当該プロセッサ管理プログラムはシステムの稼働状況を参照してこの必要プロセッサ数だけのプロセッサを特定するようにしたプロセッサ割当て方式であって、
ジョブクラスごとに、そこに属するジョブが同時使用できるプロセッサの最大個数を含むジョブクラス定義情報を記載したジョブクラス定義表をあらかじめ作成しておき、
前記ジョブ管理プログラムは、前記ジョブの定義に記述の前記ジョブクラスを基に前記ジョブクラス定義表を参照することにより求められる前記入力ジョブの前記最大個数、または前記ジョブの定義に記述のプロセッサ個数そのものを前記必要プロセッサ数として用いるようにしたことを特徴とするプロセッサ割当て方式。
【請求項２】　ジョブの定義に記述の前記プロセッサ個数そのものを前記必要プロセッサ数として用いる場合には、前記ジョブ管理プログラムは、当該プロセッサ個数がそれに対応の前記ジョブクラスの前記最大個数の範囲内であることを確認した上で前記実行要求を発生するようにした請求項１記載のプロセッサ割当て方式。
【請求項３】　前記最大個数に対応付けて組合せた各プロセッサからなるプロセッサグループを定義して、前記ジョブクラスに対応の当該プロセッサグループを前記ジョブクラス定義表に記載し、前記ジョブ管理プログラムは前記選択の対象となった前記入力ジョブに対応する当該プロセッサグループも前記プロセッサ管理プログラムに通知し、前記プロセッサ管理プログラムはこのプロセッサグループに対して前記特定を行うようにした請求項１または２記載のプロセッサ割当て方式。
【請求項４】　前記プロセッサグループの構成メンバーであるプロセッサをシステムの稼働状況に合わせて変更するようにした請求項３記載のプロセッサ割当て方式。
【請求項５】　単一の前記ジョブクラスに複数の前記プロセッサグループが対応している場合には、前記ジョブ管理プログラムの出口ルーチンによりこの中のいずれかのプロセッサグループを選択するようにした請求項３または４記載のプロセッサ割当て方式。
【発明の詳細な説明】
【０００１】
【産業上の利用分野】本発明は、マルチプロセッサシステムにおけるプロセッサ割当て方式に関し、特に多数（例えば２００台）のプロセッサを不特定多数の利用者が（複数プロセッサを用いて単一ジョブの並列処理を行うといった利用形態も含んだかたちで）共同利用するときのプロセッサ割当て方式に関する。なお、本明細書ではマルチプロセッサの全体を必要に応じて「システムプロセッサ」という。
【０００２】一般に、マルチプロセッサの利用形態としては、
・データベースマシンのように利用者は並行検索に用いるプロセッサの数や配置などに関知する必要はなく、これらについてはシステムがあらかじめ認識しているディスク上のデータベースの配置などに基づいて自律的に決定できる場合と
・科学技術計算などのように利用者（ジョブ定義者）自身がジョブの実行に必要なプロセッサをなんらかのかたちで指示することによりはじめてシステムがそれらを特定できる場合とがある。
【０００３】本発明は、後者の場合に対応し、あるジョブの実行に必要な単一または複数のプロセッサを割り当てるためのジョブ定義者の負担を軽減し、またこの割当て処理においてはシステムプロセッサの稼働状況（例えばあるプロセッサがダウンしたなど）に応じて柔軟に対応して全体としての利用効率を高めたいという要望に応えるものである。なお、以下の説明で用いる「システム」や「マルチプロセッサ」はこの後者の場合を前提としている。
【０００４】
【従来の技術】従来、複数のジョブを効率的に実行するため、各ジョブの性格（プロセッサや入出力装置の使用に関する優先権、プロセッサの使用可能時間、メモリ使用可能量、出力リスト量など）に基づいてこれらをグループ化したジョブクラスの概念が用いられている。
【０００５】そして、ジョブの定義で自らのジョブクラスを記述した各ジョブは、システムに投入されるとそのジョブクラスごとに入力ジョブ待ち行列にふりわけられ（図８参照）、当該ジョブクラスごとにあらかじめ設定された優先順位やハードウェア資源の下で実行される。
【０００６】なお、ジョブ定義ではジョブクラスとともにその範囲内での個別的条件を併せて記述することも可能であり、この場合のジョブは当該個別的条件の下で実行される。例えば、プロセッサの使用可能時間が対応のジョブクラスでは「３秒」と設定されているときにこの値を「２秒」と個別に指定することも可能である。
【０００７】このようなジョブクラスの概念を用いることにより、例えばあるジョブがそのジョブクラスで指定される時間を越えてプロセッサを使用している場合には当該ジョブの実行を打ち切ることによりハードウェア資源の無駄な使用を抑止することができる。
【０００８】また、マルチプロセッサシステムの利用形態としては、
・複数ジョブのそれぞれを別々の単一プロセッサで実行する方式

9-2

・単一ジョブを複数のプロセッサで並列処理する方式に大別されるが、後者の方式はこれまで積極的に利用されておらず、またそのときのジョブの定義では実行に必要な各プロセッサの物理番号を記述するため、ジョブ定義者にはそのジョブに必要なプロセッサを具体的に特定するといった作業が要求されることになる。

【０００９】

【発明が解決しようとする課題】このような従来のマルチプロセッサシステムでは、ジョブ定義者にとって自らのジョブの性格に合致したプロセッサを認識するといった作業が必須のものとなり、またジョブとプロセッサとの対応関係が固定的で、例えばいったん割り当てられた複数のプロセッサの中の一つがダウンしたときなどには再びジョブ定義者の方でこれに代わる新たなプロセッサを指示しないと同じ個数のプロセッサグループを構成することができないなどのため、ジョブ定義者の負担が大きく、またマルチプロセッサの運用面での柔軟性に欠けるという問題点がある。

【００１０】そこで、本発明では、いままで用いられているジョブクラスの定義情報（プロセッサや入出力装置の使用に関する優先権、プロセッサの使用可能時間、メモリ使用可能量、出力リスト量など）に、同時使用できるプロセッサの最大個数や、この最大個数に対応付けて組合せた各プロセッサからなるプロセッサグループなども記載して、ジョブの定義ではジョブクラスや当該ジョブの実行の際の必要プロセッサ数を記述すれば、すなわちそのジョブを並列処理するときに必要なプロセッサそれぞれの物理番号を具体的に記述しなくても、あとはシステムの方でこれらのジョブの定義とジョブクラスの定義情報に基づいてその実行に必要な個数だけのプロセッサを割り当てるようにすることにより、ジョブ定義者の負担を軽くするとともにプロセッサ運用面での柔軟性を高めることを目的とする。

【００１１】

【課題を解決するための手段】本発明は、ジョブ管理プログラムが、ジョブクラスごとの、同時使用できるプロセッサの最大個数などを記載したジョブクラス定義表を参照することにより、実行要求対象の入力ジョブに対応のプロセッサ個数を決定してこれをプロセッサ管理プログラムに送り、具体的なプロセッサの割当て処理はプロセッサ管理プログラムに任せるようにしたものである。

【００１２】図１は本発明の原理説明図である。図において、１は、ジョブクラス定義表であり、各ジョブクラスに属するジョブの性格（プロセッサや入出力装置の使用に関する優先権、当該ジョブが使用可能な最大限のハードウェア資源量）を示すジョブクラス定義情報が記載されている。本発明ではこのジョブクラス定義情報として、同時使用できるプロセッサの最大個数や後述のプロセッサグループ（番号）を含んでいる。２は、ジョブ管理プログラムであり、投入された各ジョブを読み取って入力ジョブキュー３にジョブクラスごとに保管し、また実行優先度にしたがって当該入力ジョブを順次選択してそれに対応のプロセッサ個数をジョブクラス定義表１を参照することにより求め当該ジョブの実行要求をこのプロセッサ個数とともにプロセッサ管理プログラム４に通知している。なお、実行が完了した入力ジョブに対する後述の所定の処理も行っている。３は、入力ジョブキューであり、入力ジョブのそれぞれをジョブクラスごとに保管している。４は、プロセッサ管理プログラムであり、ジョブ管理プログラム２から実行要求のあったジョブの実行に必要なプロセッサなどのハードウェア資源をシステムの稼働状況（プロセッサの空き状態など）に基づいて割り当てている。なお、実行完了後には、その実行ジョブが専有していたプロセッサなどの資源を開放している。５は、プロセッサグループであり、ジョブクラス定義表１に記載の最大個数に対応付けて組合せた各プロセッサからなっている（図５参照）。

【００１３】ここで、ジョブ管理プログラム２は、ジョブの定義にその実行に必要なプロセッサ個数が記述されているときにはこの値がジョブクラス定義表１に記載の最大個数の範囲内であるかどうかを確認して「範囲内」であるときには当該プロセッサ個数をプロセッサ管理プログラム４に通知し、また「範囲外」であるときには利用者に対しその旨の回答をしてプロセッサ管理プログラム４への実行要求はしない。

【００１４】そして、ジョブの定義に前記プロセッサ個数が記述されていないとき（ジョブ定義に必ず記述されている）にはジョブクラスを基にジョブクラス定義表１から対応の最大個数を求めてこれをプロセッサ管理プログラム４に通知する。

【００１５】また、プロセッサ管理プログラム４が、ある入力ジョブの実行に必要な個数のプロセッサやメモリ領域を割り当てることが出来ない稼働状況のときには、当該入力ジョブを実行待ちのジョブキューに入れ、現在使用中のプロセッサなどが空くのをまってから再度割り当てることになる。

【００１６】

【作用】本発明は、このように、並列処理を前提とするジョブの場合にもジョブの定義では単にジョブクラスまたは当該並列処理に必要となるプロセッサの個数を記述する、すなわちプロセッサの物理番号などを用いてこの個数だけのプロセッサを具体的に記述することを要しないため、ジョブ定義者のプロセッサ指定についての負担を軽減できるとともにシステムの稼働状況（プロセッサの空き状態など）に応じたプロセッサ割り当てを行うことができる。

【００１７】また、プロセッサグループを用いる場合には、ジョブ管理プログラム２は、実行要求を行う入力ジョブに対応のプロセッサグループをジョブクラス定義表１から求めて前記のプロセッサ個数や最大個数とともに

9-3

プロセッサ管理プログラム４に通知する。
【００１８】そして、この通知を受けたプロセッサ管理プログラム４は、当該入力ジョブへ、当該プロセッサグループの構成メンバーであるプロセッサの中から例えば前記プロセッサ個数だけのものを選択して割り当てる。
【００１９】このようなプロセッサグループの概念を用いることにより、メモリの実装量などの相違や入出力装置が直接付いているかどうかなどの所定の基準で各プロセッサを区分けしたものをあらかじめジョブクラスに対応させておくことが可能であり、プロセッサ管理プログラム４は実行ジョブに最適なプロセッサを効率的に特定することができる。
【００２０】さらには、実行ジョブに割り当てられるプロセッサの範囲がこのプロセッサグループによって絞りこまれているため、プロセッサの割当てやプロセッサの稼働状況の把握が簡単に行うことができる。
【００２１】また、昼間と夜間などの時間帯や実行中のジョブ数などに応じてプロセッサグループの構成メンバーを動的に変更することにより、プロセッサの利用率を向上させることができる。なお、このときの変更主体はシステム運用者、変更コマンドを入力するオペレータなどである。
【００２２】また、電源管理をこのプロセッサグループを単位として行うことにより、実行中のジョブ数が減った場合には縮退運転に移行して電力の節約化を図ることができる。なお、縮退運転は前もって決まるものではなく、このときの変更主体は主にシステム運用者である。
【００２３】
【実施例】図２～図７を参照して本発明の実施例を説明する。なお、以下の実施例では説明の便宜のためプロセッサ０からプロセッサ６の計７台のマルチプロセッサを用いることにし、またジョブクラスの数も最大で４クラスとする。
【００２４】図２は、本発明のジョブ管理およびプログラム管理の概要を示す説明図であり、11はジョブクラス定義表、12はジョブ管理プログラム、13は入力ジョブキュー、14はプロセッサ管理プログラム、15は出口ルーチン、16はプロセッサグループ変更機能プログラム、17はプロセッサ管理表、18は実行待ちのジョブキュー、19は実行中のジョブキューをそれぞれ示しており、ジョブクラス定義表11～プロセッサ管理プログラム14の作用は図１のそれぞれと基本的に同じである
【００２５】ここで、出口ルーチン15は、ジョブ管理プログラム12からの依頼により、そのときのシステム稼働状況に基づいてジョブクラス定義表11に記載の複数のプロセッサグループ（図７参照）の中から最適のものを選択する。なお、この処理はジョブ管理プログラム12が行うこともある。
【００２６】また、出口ルーチン15は、プロセッサグループの構成メンバーを変更する、例えばプロセッサ０とプロセッサ１からなるプロセッサグループにプロセッサ３を追加するといった指示をプロセッサグループ変更機能プログラム16に与えることもできる。
【００２７】そして、このプロセッサグループの構成メンバーの変更によってもジョブクラス定義表11のプロセッサグループに関する記載はそのままであり、この変更指示は前記のようにオペレータやシステム運用者なども行うことができる。
【００２８】また、プロセッサ管理表17は、各プロセッサが保有しているハードウェア資源の量と現在の使用量、例えば保有するメモリが何メガバイトでその中のいくらを実行中のジョブで使用しているといった情報に加えて、各プロセッサがどのプロセッサグループに属するかを示すプロセッサグループ定義情報を含んでおり、さらには各プロセッサの稼働状況を管理している。
【００２９】なお、各プロセッサは複数のプロセッサグループに属することもでき、また前記のように例えば出口ルーチン15からの指示により各プロセッサが所属するプロセッサグループをシステム運用状況に対応して変更することも可能である。
【００３０】また、ジョブ管理プログラム12から実行要求のあった各入力ジョブの中、その実行に必要なハードウェア資源を割り当てることができたものは実行中のジョブキュー19に、またこれらのハードウェア資源を割り当てることができないものは実行待ちのジョブキュー18にそれぞれ保管される。
【００３１】図３は、ジョブ管理プログラム12の処理手順を示す説明図であり、その内容は次のようになっている。
入力される各ジョブの定義情報を解析してそれぞれがどのジョブクラスに属するかを求め、これらの入力ジョブをジョブクラス別に入力ジョブキュー13に入れるとともに入力ジョブ数をカウントしていく。
実行優先度の最も高いジョブクラスの入力ジョブを入力ジョブキュー13から取り出して、次のステップに進む。なお、各ジョブクラスの実行優先度はジョブクラス定義表11に記載するかジョブの定義で記述するかのいずれでもよい。
前のステップで取り出した入力ジョブを実行すべきかどうかを、ジョブの多重度を考慮して判断し、「YES」の場合はステップに進み、「NO」の場合は次のステップに進む。なお、ジョブの多重度はジョブクラスごとに決まっているものでジョブクラス定義表11に記載されている。
当該入力ジョブの属するジョブクラスの実行優先度を例えば第２位下げて、ステップに戻る。
ジョブクラス定義表11を参照して当該ジョブのジョブクラスで指定のプロセッサグループを求めて、次のステップに進む。
当該プロセッサグループが複数であるかどうかを判断

9-4

し、「YES 」の場合は次のステップに進み、「NO」の場合はステップに進む。
出口ルーチン15に複数のプロセッサグループの中のいずれを用いるかの確定を依頼し、その回答を確認して、次のステップに進む。なお、この確定はジョブ定義情報（ジョブの実行に必要なプロセッサの個数、実行時間の予測値など）やシステム稼働情報（プロセッサの空き状態など）を基に行われる。
当該入力ジョブの実行要求を、対応するプロセッサグループ番号、実行に必要なプロセッサの個数や実行優先度などの情報とともにプロセッサ管理プログラム14に通知して、ステップに戻る。
プロセッサ管理プログラム14から入力ジョブの実行完了通知を受けることに、当該入力ジョブを入力ジョブキュー行列13から除くとともに入力ジョブ数を「1」だけ減じる。
【００３２】そして、以上のステップ～ステップは、
・ジョブの受付処理であるステップ
・ジョブの実行要求処理であるステップ～ステップ
・ジョブの実行完了後の処理であるステップ
に大別され、これらの処理単位は相互に独立したものとなっており、例えばステップやステップの終了後でもジョブの入力があればステップの処理が始まるようになっている。また、ステップの後で入力ジョブ数が「0」の状態が継続するときはジョブ管理プログラム12の処理は一応の終了となる。
【００３３】図４は、プロセッサ管理プログラム14の処理手順を示す説明図であり、その内容は次のようになっている。
' 前記ステップで実行要求のあった各入力ジョブをそれぞれの実行優先度を考慮して実行待ちのジョブキュー18に繋ぐ。
' 実行優先度の最も高い入力ジョブを実行待ちのジョブキュー18から取り出して、次のステップに進む。
' 当該入力ジョブの実行に必要な量のプロセッサやメモリなどを割り当てることができるかどうかを判断し、「YES」の場合は次のステップに進み、「NO」の場合はステップ' に進む。なお、プロセッサの割当ては前記ステップのプロセッサグループの各プロセッサに対して行われる。
' 当該入力ジョブを実行して実行中のジョブキュー19に繋ぐとともに、当該入力ジョブに割り当てた資源の量をプロセッサ管理表17に記入してこの内容を更新して、ステップ' に戻る。
' 当該入力ジョブの実行優先度を下げて、ステップ' に戻る。
' 任意の入力ジョブの実行完了にともない、当該入力ジョブが専有していた資源を解放してプロセッサ管理表17の内容を更新するとともに、ジョブ管理プログラム12に当該入力ジョブの実行完了を通知する。
【００３４】以上のステップ' ～ステップ' はジョブ管理プログラム12の場合と同じように、
・ジョブの受付処理であるステップ'
・ジョブの実行処理であるステップ' ～ステップ'
・ジョブの実行完了後の処理であるステップ'
に大別され、これらの処理単位もまた相互に独立したものとなっている。
【００３５】なお、前記ステップ' やステップ' における実行優先度は、例えば各ジョブクラス間の実行優先度は保持したままで同じジョブクラスの中についてはジョブ管理プログラム12から実行要求のあった順序としている。
【００３６】また、実行要求のあったすべてのジョブについてその実行完了がジョブ管理プログラム12に通知されると、前記ステップの処理後の入力ジョブ数が「0」となってプロセッサ管理プログラム14の方でもこの値を確認することにより自らの一応の終了を認識できる。
【００３７】図６～図７は、本発明のマルチプロセッサの運用例を示す説明図であり、図５ではジョブの実行予定、ジョブクラスの定義およびプロセッサグループの定義の一例を示している。
【００３８】すなわち、ジョブの実行予定は昼間と夜間とで区別され、例えば昼間の場合には最大４並列のプロセッサを使用するジョブと単一プロセッサを使用する非並列ジョブとを実行するようになっている〔図５(a) 参照〕。
【００３９】また、例えばジョブクラスＡには最大４並列のプロセッサを使用して実行されるジョブが属し、このジョブクラスに対応のプロセッサグループとして「プロセッサグループ0」が定義されている〔図５(b) 参照〕。
【００４０】また、例えば「プロセッサグループ0」は４並列ジョブ用として定義されその構成メンバーとして「プロセッサ0」～「プロセッサ３」が当てられている〔図５(c) 参照〕。なお、非並列ジョブ用の「プロセッサグループ２」と７並列ジョブ用の「プロセッサグループ３」との構成メンバーを重複させることにより、夜間、６並列以下の並列ジョブと非並列ジョブとの同時実行を可能にしている。
【００４１】そして、図５の運用環境の下では、昼間は、ジョブクラスの実行優先度を「ジョブクラスＡ－ジョブクラスＢ－ジョブクラスＣ・ジョブクラスD」の順、すなわちジョブクラスＡのそれを第１位にしておき、夜間になるとコマンドなどでこの実行優先度を「ジョブクラスD－ジョブクラスC－ジョブクラスＡ・ジョブクラスＢ」の順に変更する。
【００４２】このようにすることにより、昼間はジョブクラスＡ（最大４並列）のジョブとジョブクラスＢ（非

9-5

並列）のジョブがそれぞれ別々のプロセッサグループで実行され、また夜間になると、ジョブクラスD（最大７並列）のジョブがジョブクラスC（非並列）のジョブに優先するかたちで実行される。

【００４３】なお、夜間の場合、ジョブクラスCの非並列ジョブは、ジョブクラスDの並列ジョブが６並列以下のときにはこの並列ジョブが使用していないプロセッサで実行され、またジョブクラスDの並列ジョブが７並列のときにはこの並列ジョブが終了してから実行されることになる。

【００４４】図６では、プロセッサグループのメンバー構成を各プロセッサの稼働状況に合わせて変更する場合を示している。すなわち、図５(c)の、４並列ジョブ用のプロセッサグループ０と非並列ジョブ用のプロセッサグループとが稼働しているときのある時点で、

・４並列ジョブのすべての実行が終了し、かつ実行待ちの非並列ジョブが沢山あるときには例えばコマンドを用いて「プロセッサグループ０」の構成メンバーを「プロセッサグループ１」に追加するようにし、

・ある４並列ジョブの実行中にプロセッサ３がダウンしたときには例えばコマンドを用いて「プロセッサ４」を「プロセッサグループ０」に追加することにより、実行待ちの非並列ジョブや４並列ジョブを即座に実行することができる。

【００４５】この運用によれば、ジョブクラス定義表11のプロセッサグループ番号は初期の記載のままでそのプロセッサグループの構成メンバーが各プロセッサの稼働状況に合わせて動的に変更されることになる。

【００４６】図７では、一つのジョブクラスに複数のプロセッサグループを対応させる場合を示している。すなわち、最大４並列のプロセッサまで使用可能なジョブが属するジョブクラスＡには４並列用の「プロセッサグループ０」のみを、また単一プロセッサを使用する非並列ジョブが属するジョブクラスＢには非並列用の「プロセッサグループ１」に加えて「プロセッサグループ０」も対応させている。

【００４７】この運用の際には、ジョブクラスＢのジョブの実行に先だってどちらのプロセッサグループを用いるかを決定する必要があり、この決定は例えば出口ルーチン15がシステムの稼働状況を参照しながら、

・プロセッサグループ１に空きプロセッサがあれば当該グループを選択し、

・プロセッサグループ１に空きプロセッサがなく、プロセッサグループ０に空きプロセッサがあれば、すなわちその時点で４並列ジョブが実行されていなければプロセッサグループ０を選択し、

・さもなければこれらのいずれかの状況が発生するのを待って対応のプロセッサグループを選択するといったかたちで行っている。

【００４８】そして、この運用によれば、４並列ジョブが実行されていないときには、図６のようなプロセッサグループの構成メンバーの変更作業を行わなくとも非並列ジョブを４並列用プロセッサで実行することができる。

【００４９】

【発明の効果】本発明は、このように、従来のジョブクラス定義に、各ジョブクラスに属するジョブが並列使用可能なプロセッサの最大個数などの情報も併せて記載し、マルチプロセッサシステムのユーザの方では投入するジョブのジョブクラスや当該ジョブの実行に必要なプロセッサ個数を指示するだけで後はシステムがジョブの実行に必要な単一または複数のプロセッサを前記情報に基づいて割り当てるようにしているため、ジョブ実行時の使用プロセッサ指示についてのジョブ定義者の負担を軽減することができる。

【００５０】さらには、プロセッサグループの概念を用い、このプロセッサグループの構成メンバーをシステムの稼働状況に合わせて（ジョブクラス定義におけるプロセッサグループの記載はそのままで）動的に変更し、また単一のジョブクラスに複数のプロセッサグループを対応させておきいずれを選択するかは稼働状況に応じて動的に行うようにしているため、プロセッサ運用面での柔軟性を高めてマルチプロセッサ全体の効率的な使用が可能となる。

【図面の簡単な説明】

【図１】本発明の、基本構成図である。

【図２】本発明の、ジョブ管理およびプログラム管理の概要を示す説明図である。

【図３】本発明の、ジョブ管理プログラムの処理手順を示す説明図である。

【図４】本発明の、プロセッサ管理プログラムの処理手順を示す説明図である。

【図５】本発明の、ジョブの実行予定、ジョブクラスの定義およびプロセッサグループの定義の一例を示す説明図である。

【図６】本発明の、プロセッサグループのメンバー構成を各プロセッサの稼働状況に合わせて変更する場合を示す説明図である。

【図７】本発明の、一つのジョブクラスに複数のプロセッサグループを対応させる場合を示す説明図である。

【図８】一般的な、入力ジョブの保管の様子を示す説明図である。

【符号の説明】

図１において、

１・・・ジョブクラス定義表

２・・・ジョブ管理プログラム

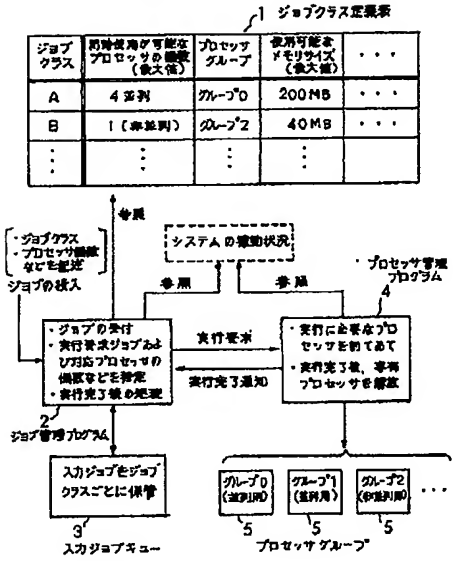
３・・・入力ジョブキュー

４・・・プロセッサ管理プログラム

５・・・プロセッサグループ

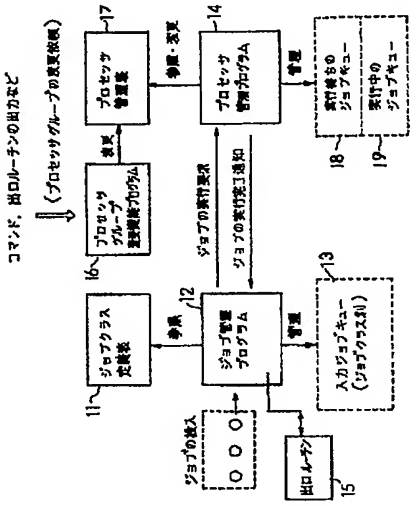
【図１】

本発明の基本構成



【図２】

本発明のジョブ管理およびプログラム管理の概要を示す説明図



【図７】

本発明の 一つのジョブクラスに複数のプロセッサグループを対応させる場合を示す説明図

(a) ジョブクラスの定義

ジョブクラス	プロセッサの並列状況	プロセッサグループ
A	最大4並列	0
B	非並列	0，1

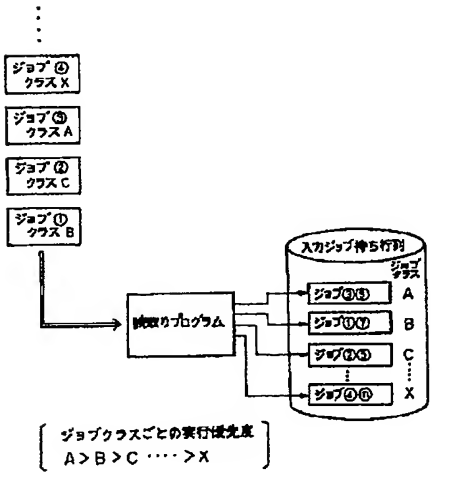
(b) プロセッサグループの定義

	プロセッサグループ0	プロセッサグループ1
プロセッサ0	●	
プロセッサ1	●	
プロセッサ2	●	
プロセッサ3	●	
プロセッサ4		●
プロセッサ5		●
プロセッサ6		●

〔プロセッサグループ0は 4並列ジョブ用（最大）
プロセッサグループ1は 非並列ジョブ用〕

【図８】

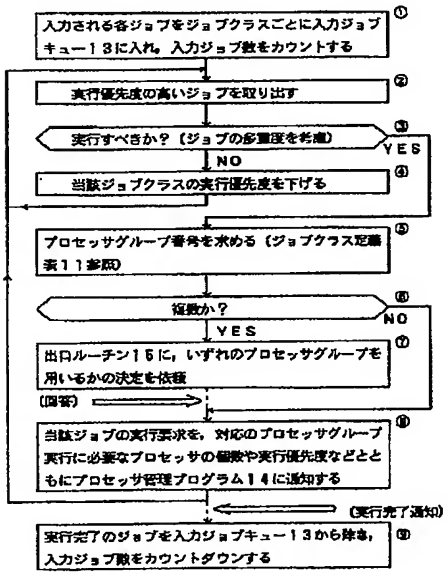
一般的な入力ジョブの保管の様子を示す説明図



9-7

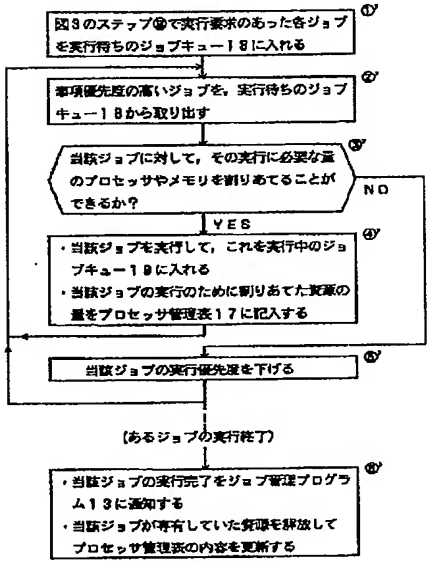
【図３】

本発明の，ジョブ管理プログラムの処理手順を示す説明図



【図４】

本発明の，プロセッサ管理プログラムの処理手順を示す説明図



9-8

【図５】

本発明の ジョブの実行予定，ジョブクラスの定義
およびプロセッサグループの定義の一例を示す説明図

(a) ジョブの実行予定

昼 間	最大４並列のジョブと非並列ジョブを実行
夜 間	最大７並列のジョブと非並列ジョブを実行

(b) ジョブクラスの定義

ジョブクラス	定義内容	
A	最大4並列	プロセッサグループ 0
B	非並列	プロセッサグループ 1
C	非並列	プロセッサグループ 2
D	最大4並列	プロセッサグループ 3

(c) プロセッサグループの定義

（プロセッサグループ0は4並列ジョブ用（最大）
プロセッサグループ1は非並列ジョブ用
プロセッサグループ2は非並列ジョブ用
プロセッサグループ3は7並列ジョブ用（最大））

	プロセッサグループ0	プロセッサグループ1	プロセッサグループ2	プロセッサグループ3
プロセッサ0	●		●	●
プロセッサ1	●		●	●
プロセッサ2	●		●	●
プロセッサ3	●		●	●
プロセッサ4		●	●	●
プロセッサ5		●	●	●
プロセッサ6		●	●	●

【図６】

本発明の，プロセッサグループのメンバー構成を各プロセッサの
稼動状況に合わせて変更する場合を示す説明図

（プロセッサグループ0は4並列ジョブ用（最大）
プロセッサグループ1は非並列ジョブ用）

	プロセッサグループ0	プロセッサグループ1
プロセッサ0	●	
プロセッサ1	●	
プロセッサ2	●	
プロセッサ3	●	
プロセッサ4		●
プロセッサ5		●
プロセッサ6		●

（変更）

（4並列ジョブの実行をすべて完了したとき）

	プロセッサグループ0	プロセッサグループ1
プロセッサ0	●	●
プロセッサ1	●	●
プロセッサ2	●	●
プロセッサ3	●	●
プロセッサ4		●
プロセッサ5		●
プロセッサ6		●

（プロセッサ3がダウンしたとき）

	プロセッサグループ0	プロセッサグループ1
プロセッサ0	●	
プロセッサ1	●	
プロセッサ2	●	
プロセッサ3		
プロセッサ4	●	●
プロセッサ5		●
プロセッサ6		●

9-9